CMPT 370 - 1998

A: Multiple Choice (5 marks). Choose the best available answer for each of the following five questions (1 mark each).

1. The two key contributors to the early design and implementation of the UNIX operating system were:
(a) IBM and AT&T
(b) IBM and Microsoft
(c) AT&T and the University of California at Berkeley
(d) Dennis Ritchie and Ken Thompson
(e) Dennis Ritchie and Bill Gates
(f) Brian Kernighan and Dennis Ritchie

2. When programming in the Bourne shell, the conventional way to indicate "normal" termination is with an exit status of:
(a) 0
(b) 1
(c) -1
(d) any positive integer larger than 1
(e) $?
(f) the process id (PID) of the executing process when it terminates
(g) the PID of the shell (parent) process itself

3. Upon which inode number in the inode list is the root ("/") of the UNIX file system usually established?
(a) Inode 0
(b) Inode 1
(c) Inode 2
(d) Inode 3
(e) Inode 4
(f) None of the above

4. The maximum number of open file descriptors that a UNIX process can have at any one time (in a traditional UNIX system with default settings of kernel parameters) is:
(a) 3 (standard input, standard output, and standard error)
(b) 4
(c) 8
(d) 10
(e) 16
(f) 32
(g) 64
(h) None of the above

5. The return value from a successful fork() system call is:
(a) always 0
(b) always greater than 0
(c) never less than 0
(d) the PID number of the child process that was created
(e) different in the parent process than in the child process
(f) dependent upon the version of UNIX on which it is run

B: Short Answer (12 marks). Write a clear concise answer to each of the following questions (3 marks each).

6. What is a "superblock"? What information does it contain? What purpose does it serve?

Block at start of file system (after bootstrap) that describes the file system characteristics
- max # of inodes
- max # of data blocks
- list of inodes
- list of free data blocks

7. List three of the most important enhancements (i.e., improvements) that the "Fast File System" contributed to UNIX.

(i) variable length file names (directory storage structure)

(ii) Efficiency of Storage? more specific.

(iii) performance optimization algorithm?

-2

8. What is the difference between the "data segment" and the "stack segment" of a UNIX process? Define each, and clarify their differences. Be as specific as possible.

- data segment is an block of memory which is allocated @ start for user variables/data (static allocation)
- Stack is a block of memory used for dynamic storage during program execution

-2

9. What is a "read-only" shell variable? What is such a variable usually used for? Give an example.

a variable that the script can retrieve but not set the value of.

ex $SHELL $TERM $HOST

each are environment variable provided by the system.

8/12

C: Processes (14 marks). Write a clear, concise answer for each of the following questions.

10. The UNIX command "ps –aux" shows much more information than the "ps" command alone (i.e., "ps" without any command line options). List three examples of information that is available from the output of "ps –aux" that is not available with the output from just "ps". Be specific. (4 marks)

(i) lists all processes instead of just current user

(ii) Resident Size of program

(iii) Actual Size of program similar -1

11. List any five of the typical attributes of a UNIX process. Describe each briefly in terms of what it is, and what role it plays in the execution, management, or control of the process. Be specific. (10 marks)

(i) PID process id - uniquely identifies the process.

(ii) UID user id - identifies access permissions based on the user id or creator of the process

(iii) GID group id - identifies access permissions based on the group id

related -2

(iv) EUID effective user id - identifies access permissions that the process has based not on its real user id but the user id stored in EUID

(v) status

- running
- asleep
- Zombie
- waiting

describes what the process is doing.

vi) PPID parent process id - stores unique id of the parent which created the current process

11/14

D: File System (14 marks). Write a clear concise answer to each of the following questions.

12. An inode in the UNIX file system stores three different timestamp values. Define these timestamps, and state briefly what each of these timestamps represents. Be as precise as possible. (5 marks)

(i) file [redacted]/modification time - Time the file was last written to

(ii) [redacted] last access time. - time the file was last read.

(iii) inode status change time - Time that an attribute of the inode (ex: # of links) changed

13. The UNIX file system can run "out of space" in two distinctly different ways. Describe each of these two possibilities, and indicate how this might possibly occur in practice. (4 marks)

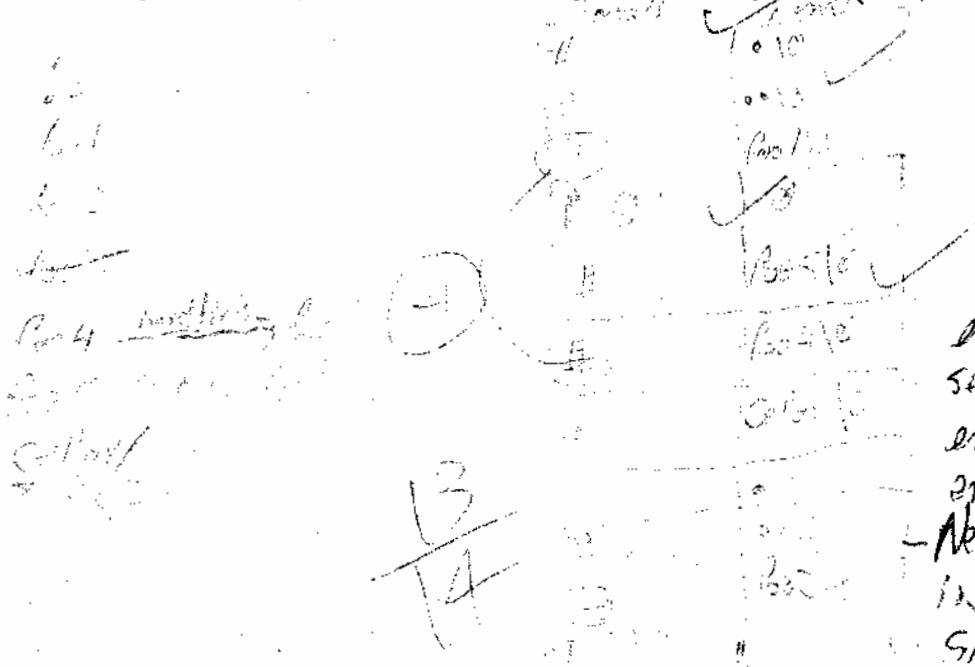
(i) [redacted] all inodes are allocated.
- if file system has many files that are either small or have large "holes" in them.
- few data blocks per inode.

(ii) all data blocks are allocated.
- if file system has many large files
- a lot of data blocks per inode.

14. Suppose that a directory /usr/ming/fun in the UNIX file system "contains" exactly three files, which are named "foo1", "foo2", and "foo3", respectively, to represent the order of their creation. Each is a regular file, and each file is distinct in terms of its size and contents. Now the owner of the directory performs the following UNIX command in this directory: "ln foo1 foo4 ; /bin/rm foo3 ; cp foo4 foo5 ; mkdir cellar ; mv foo2 cellar ; ls –l". Draw a diagram to illustrate the complete structure, order, and contents of the directory table for /usr/ming/fun, assuming the standard algorithms and 16-byte storage technique for directory entries as used in the original UNIX file system. (5 marks)

only 1 of answer

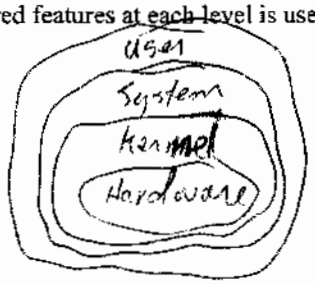
2 byte inode#
14 byte name



entries are stored sequentially. when an entry is removed the table entry stores a value of 0.
- new entries are placed in the first available space.

E: General (5 marks).

15. The UNIX operating system and programming environment offers a rich mixture of tools and utilities for the systems programmer, some of which are available at the shell (i.e., command-line) level, some of which are available at the system library level, and some of which are available as (kernel) system calls. Clarify what is meant by each of these "levels" (i.e., command-line, library, and system call), giving an example of each, and explain why having an assortment of offered features at each level is useful.



1. command line
   >ls

2. library
   in a C program
   _exit(x);

3. System call
   ~~A~~ in a C program
   ~~B~~ close(fd);

- Shell or command line level is the typical user level - application level
- System library level is where functions/programs that interface between the kernel via system calls and the user level.
- Kernel level interfaces actual hardware with the rest of the system

*** THE END ***

- each level from the kernel upwards, presents more robustness and ability and complexity of interaction
  ~~example~~